interpretive language execution
Terms used: interpretive language execution

Found 403 of 239,726

| Sort results by | relevance ▼ | ◆ Save results to a Binder | Refine these results with Advanced Search |
| Display results | expanded form ▼ | ☐ Open results in a new window | Try this search in The ACM Guide |

Results 1 - 20 of 403   Result page: 1   2   3   4   5   6   7   8   9   10   next   >>

1   TOOLS: a unifying approach to object-oriented language interpretation
    K. Koskimies, J. Paakki
    July 1987 SIGPLAN '87: Papers of the Symposium on Interpreters and interpretive techniques
    Publisher: ACM

    Full text available: 📄 pdf(914.68 KB)   Additional Information: full citation, abstract, index terms

    The object-oriented paradigm is applied to the interpreting of programming languages. An intermediate representation of a program is created as a collection of objects representing various entities in the conceptual world of the source language. These ...

2   A methodology for the design of application specific instruction set processors (ASIP) using the machine description language LISA
    Andreas Hoffmann, Oliver Schliebusch, Achim Nohl, Gunnar Braun, Oliver Wahlen, Heinrich Meyr
    November 2001 ICCAD '01: Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design
    Publisher: IEEE Press

    Full text available: 📄 pdf(913.02 KB)   Additional Information: full citation, abstract, references, cited by, index terms

    The development of application specific instruction set processors (ASIP) is currently the exclusive domain of the semiconductor houses and core vendors. This is due to the fact that building such an architecture is a difficult task that requires expertise ...

3   Shade: a fast instruction-set simulator for execution profiling
    Bob Cmelik, David Keppel
    May 1994 SIGMETRICS '94: Proceedings of the 1994 ACM SIGMETRICS conference on Measurement and modeling of computer systems
    Publisher: ACM

    Full text available: 📄 pdf(1.28 MB)   Additional Information: full citation, abstract, references, cited by, index terms

    Tracing tools are used widely to help analyze, design, and tune both hardware and software systems. This paper describes a tool called Shade which combines efficient instruction-set simulation with a flexible,

extensible trace generation capability. ...

4 Cint: a RISC interpreter for the C programming language

J. W. Davidson, J. V. Gresh
July 1987 SIGPLAN '87: Papers of the Symposium on Interpreters and
        interpretive techniques
Publisher: ACM

Full text available: pdf(790.29 KB) Additional Information: full citation, abstract, cited by,
                                           index terms

Cint is an interpretation system for the C programming language. Like
most interpretation systems, it provides "load and go" type execution as
well as enhanced debugging and performance analysis tools. Cint
consists of two phases--a translator ...

5 Interpretive execution of real-time control applications

Mary S. Adix, Henrik A. Schutz
March 1976 ACM SIGMINI Newsletter, Volume 2 Issue 2
Publisher: ACM

Full text available: pdf(86.15 KB) Additional Information: full citation, abstract

Interpretive execution has often been regarded as too slow for real-time
control applications. Assembly language implementations, however, may
exhaust available memory long before running out of machine cycles.
For such applications, interpretation of ...

6 The PL/EXUS language and virtual machine

Gary A. Sitton, Thomas A. Kendrick, A. Gil Carrick, Jr.
November 1973 Proceedings of the ACM-IEEE symposium on High-level-
        language computer architecture
Publisher: ACM

                    Additional Information: full citation, abstract,
Full text available: pdf(650.16 KB)   references, cited by, index
                                          terms

This paper describes a high level general purpose language which
evolved from another high level systems programming language. As
well, the compiler, pseudocode, and virtual machine are discussed in
some detail. The new language is a powerful PL/1 dialect, ...

7 A practical and flexible flow analysis for higher-order languages

J. Michael Ashley, R. Kent Dybvig
July 1998 ACM Transactions on Programming Languages and Systems
        (TOPLAS), Volume 20 Issue 4
Publisher: ACM

                    Additional Information: full citation, abstract,
Full text available: pdf(319.36 KB)   references, cited by, index
                                          terms

A flow analysis collects data-flow and control-flow information about
programs. A compiler can use this information to enable optimizations.
The analysis described in this article unifies and extends previous work
on flow analysis for higher-order languages ...

Keywords: abstract interpretation, higher-order languages

8 An overview of nonprocedural languages
Burt M. Leavenoworth, Jean E. Sammet
April 1974 ACM SIGPLAN Notices, Volume 9 Issue 4
Publisher: ACM

Full text available: 📄 pdf(1.25 MB)   Additional Information: full citation, abstract, references, cited by, index terms

This paper attempts to describe some of the basic characteristics and issues involving the class of programming languages commonly referred to as "nonprocedural" or "very high level". The paper discusses major issues such as terminology, ...

9 A generator for language-specific debugging systems
R. Bahlke, B. Moritz, G. Snelting
July 1987 SIGPLAN '87: Papers of the Symposium on Interpreters and
          interpretive techniques
Publisher: ACM

Full text available: 📄 pdf(583.68 KB)   Additional Information: full citation, abstract, cited by, index terms

We present a system which generates interactive high-level debugging systems from formal language definitions. The language definer has to specify a denotational semantics augmented with a formal description of the language specific debugging facilities. ...

10 An abstract machine for tabled execution of fixed-order stratified logic programs
Konstantinos Sagonas, Terrance Swift
May 1998 ACM Transactions on Programming Languages and Systems
          (TOPLAS), Volume 20 Issue 3
Publisher: ACM

Additional Information: full citation, abstract, references, cited by, index terms, review

Full text available: 📄 pdf(602.38 KB)

SLG resolution uses tabling to evaluate nonfloundering normal logic programs according to the well-founded semantics. The SLG-WAM, which forms the engine of the XSB system, can compute in-memory recursive queries an order of magnitude faster ...

Keywords: SLG, WAM, memoing, prolog, stratification theories, tabling

11 Systematically derived instruction sets for high-level language support
Pradip Bose, B. R. Rau, M. S. Schlansker
April 1982 ACM-SE 20: Proceedings of the 20th annual Southeast regional
          conference
Publisher: ACM

Full text available: 📄 pdf(729.75 KB)   Additional Information: full citation, abstract, references, cited by

Conventional machine-languages (instruction sets) were not designed with high-level languages (HLLs) in mind. The resulting semantic gap is known to cause significant inefficiencies in program representation and execution time. Direct interpretation ...

Keywords: compilation, directly interpretable languages, high-level languages, instruction set design, interpretation, semantic gap, space-time efficiency, syntax and semantics

12 Interpretive execution of real-time control applications
    Mary S. Adix, Henrik A. Schutz
    April 1976 ACM SIGPLAN Notices, Volume 11 Issue 4
    Publisher: ACM

    Additional Information: full citation, abstract,
    Full text available: pdf(646.96 KB)   references, cited by, index
                                                             terms

    Interpretive execution has often been regarded as too slow for real-time
    control applications. Assembly language implementations, however, may
    exhaust available memory long before running out of machine cycles.
    For such applications, interpretation of ...

13 Determining average program execution times and their variance
    V. Sarkar
    July 1989 ACM SIGPLAN Notices, Volume 24 Issue 7
    Publisher: ACM

    Full text available: pdf(1.18 MB) Additional Information: full citation, abstract, references,
                                                             cited by, index terms

    This paper presents a general framework for determining average
    program execution times and their variance, based on the program's
    interval structure and control dependence graph. Average execution
    times and variance values are computed using frequency ...

14 Architecture description language (ADL)-driven software toolkit
    generation for architectural exploration of programmable SOCs
    Prabhat Mishra, Aviral Shrivastava, Nikil Dutt
    July 2006 ACM Transactions on Design Automation of Electronic
              Systems (TODAES), Volume 11 Issue 3
    Publisher: ACM

    Full text available: pdf(1.07 MB) Additional Information: full citation, abstract, references,
                                                             cited by, index terms

    Advances in semiconductor technology permit increasingly complex
    applications to be realized using programmable systems-on-chips
    (SOCs). Furthermore, shrinking time-to-market demands, coupled with
    the need for product versioning through software modification ...

    Keywords: Architecture description language, design space exploration,
    embedded processor, programmable architecture, retargetable
    compilation

15 A minisystem programming language
    Robert Lechner, William Stallings
    August 1973 ACM'73: Proceedings of the annual conference
    Publisher: ACM

    Additional Information: full citation, abstract,
    Full text available: pdf(544.11 KB)   references, cited by, index
                                                             terms

    TRAIL is a block-structured language and programming system for the
    development of programming support systems and translators for

> problem-oriented languages on minicomputers. The programming
> system includes an interpreter for an intermediate language ...

16 The PL/EXUS language and virtual machine
Gary A. Sitton, Thomas A. Kendrick, A. Gil Carrick, Jr.
November 1973 ACM SIGPLAN Notices, Volume 8 Issue 11
Publisher: ACM

Additional Information: full citation, abstract,
Full text available: 🔲 pdf(650.16 KB)   references, cited by, index
                                                                                terms

> This paper describes a high level general purpose language which
> evolved from another high level systems programming language. As
> well, the compiler, pseudocode, and virtual machine are discussed in
> some detail. The new language is a powerful PL/1 dialect, ...

17 A language and model for computer design
N. G. Denil
July 1966 Communications of the ACM, Volume 9 Issue 7
Publisher: ACM
Full text available: 🔲 pdf(654.43 KB) Additional Information: full citation

18 A tutoring system for parameter passing in programming languages
Harsh Shah, Amruth N. Kumar
June 2002 ITiCSE '02: Proceedings of the 7th annual conference on
                Innovation and technology in computer science education
Publisher: ACM

Additional Information: full citation, abstract,
Full text available: 🔲 pdf(176.86 KB)   references, cited by, index
                                                                                terms

> We have developed a tutoring system for the parameter passing
> mechanisms discussed in a typical *Comparative Programming Languages*
> course, viz., value, result, value-result, reference and name. The tutor
> helps students better understand these parameter ...
>
> Keywords: active learning, evaluating educational software, online
> learning, parameter passing mechanisms in programming languages,
> problem-solving, web-based tutors

19 Optimizing strategies for telescoping languages: procedure strength
    reduction and procedure vectorization
Arun Chauhan, Ken Kennedy
June 2001 ICS '01: Proceedings of the 15th international conference on
                Supercomputing
Publisher: ACM

Additional Information: full citation, abstract,
Full text available: 🔲 pdf(193.02 KB)   references, cited by, index
                                                                                terms

> At Rice University, we have undertaken a project to construct a
> framework for generating high-level problem solving languages that can
> achieve high performance on a variety of platforms. The underlying
> strategy, called *telescoping languages*, builds ...

Keywords: Matlab, automatic differentiation, high-level languages, high-performance computing, partial evaluation, procedure specialization, reduction in strength, scripts, specialization, telescoping languages, vectorization

20 Tagless staged interpreters for typed languages
Emir PašaliE, Walid Taha, Tim Sheard
September 2002 ACM SIGPLAN Notices,   Volume 37 Issue 9
Publisher: ACM

Additional Information: full citation, abstract,
Full text available: ⬛ pdf(203.54 KB)   references, cited by, index
                                                                        terms

Multi-stage programming languages provide a convenient notation for explicitly staging programs. Staging a definitional interpreter for a domain specific language is one way of deriving an implementation that is both readable and efficient. In an untyped ...

Keywords: calculus of constructions, definitional interpreters, domain-specific languages, multi-stage programming

Results 1 - 20 of 403   Result page: 1   2   3   4   5   6   7   8   9   10   next   >>